INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

NOMBRES:

PADILLA MATIAS CRISTIAN MICHEL

SAUCILLO GONZÁLEZ JESSE OBED

GRUPO:

4BM1

TRABAJO:

Practica 6

"Representación mediante marcos"

MATERIA:

Fundamentos de Inteligencia Artificial

FECHA

14 de noviembre del 2023

PRACTICA 5. Razonamiento basado en reglas

RESUMEN

El trabajo presenta la implementación de una representación mediante marcos en Prolog para modelar una taxonomía de animales. Se utiliza una estructura de datos compleja llamada "marco" que representa situaciones estereotipadas. Cada marco tiene casillas donde se almacena información, y se organiza jerárquicamente en una base de conocimientos.

Se proporcionan marcos de representación en Prolog, junto con una base de conocimientos que abarca categorías como mamíferos, aves, reptiles, anfibios y peces, así como ejemplos específicos de cada categoría.

El trabajo incluye la implementación de una interfaz gráfica en Python utilizando Tkinter y PySWIP. La interfaz permite consultar información sobre especies seleccionadas, mostrando propiedades y, en algunos casos, imágenes asociadas. Se proporciona una lista desplegable para elegir la especie y un botón para realizar la consulta.

El código de la interfaz utiliza bibliotecas como Tkinter para la interfaz gráfica, PySWIP para la integración con Prolog y PIL para mostrar imágenes. Además, se incluyen reglas de inferencia en Prolog para consultar propiedades y relaciones entre clases.

INTRODUCCIÓN

Prolog es un lenguaje de programación lógica comúnmente utilizado para aplicaciones de inteligencia artificial. Cuando se trata de marcos de modelado, Prolog en sí puede considerarse un marco de modelado debido a su capacidad para representar y razonar sobre relaciones y reglas complejas. Además, existen bibliotecas y herramientas disponibles para Prolog que pueden ayudar en el modelado, como la biblioteca PySWIP para interconectar Prolog con Python.

Dentro del marco, existen conocimientos procedimentales que se refieren a: cómo utilizar el marco, qué se espera que suceda a continuación, así como el conjunto de

acciones que se deben realizar tanto si las expectativas se cumplen como si estas fallan. Los marcos organizan los conocimientos del dominio en árboles, también llamados jerarquías, o en grafos, ambos construidos por especialización de conceptos generales en conceptos más específicos.

PySWIP: PySWIP es una biblioteca de Python que proporciona un puente entre Prolog y Python, lo que permite una integración perfecta del código de Prolog dentro de los programas de Python. Permite la ejecución de consultas de Prolog y el intercambio de datos entre Prolog y Python, lo que la convierte en una herramienta valiosa para aplicaciones de programación lógica y de inteligencia artificial.

DESARROLLO

El código en Prolog proporciona una implementación de la representación de marcos para modelar una taxonomía de animales.

Definición de Marcos:

Los marcos se definen como estructuras de datos complejas que representan situaciones estereotipadas.

Cada marco tiene un nombre de clase, una relación de subclase y una lista de propiedades.

```
%reino
frame(animal, subclase_de(objeto), propiedades([puede(sentir), puede(respirar)])).

%tipos
frame(mamifero, subclase_de(animal), propiedades([puede(mamar), respira(aire)])).
frame(ave, subclase_de(animal), propiedades([poseen(alas), tienen(pico)])).
frame(reptil, subclase_de(animal), propiedades([tienen(sangre_fria), piel(seca),cuerpo(alargado)])).
frame(anfibio, subclase_de(animal), propiedades([son(ectotermos),pasan_por(metamorfosis)])).
frame(pez, subclase_de(animal), propiedades([son(oviparos),tienen(aletas),poseen(escamas)])).
```

Jerarquía de Clases:

Se establece una jerarquía de clases mediante la relación de subclase_de. Por ejemplo, "mamífero" es una subclase de "animal".

```
%reino
frame(animal, subclase_de(objeto), propiedades([puede(sentir), puede(respirar)])).

%tipos
frame(mamifero, subclase_de(animal), propiedades([puede(mamar), respira(aire)])).
```

Instancias y Propiedades Específicas:

Se definen instancias específicas de las clases con propiedades particulares.

```
%Para consultar todas las clases representadas en los frames
clases(L):-findall(X,frame(X,subclase_de(_),propiedades(_)),L).
```

Reglas de Inferencia:

Se proporcionan reglas de inferencia para determinar la subclase de una clase y consultar propiedades.

```
%Reglas de inferencia
subc(C1,C2):- frame(C1,subclase_de(C2),_).
subc(C1,C2):- frame(C1,subclase_de(C3),_),subc(C3,C2).
subclase(X):-frame(X,subclase_de(_),_).
```

Consultas:

Se incluyen consultas para obtener subclases, superclases y propiedades de clases específicas.

```
%Para consultar propiedades use: propiedadesc(luciernaga, L).
propiedadesc(objeto,_):-!.
propiedadesc(X,Z):-frame(X,subclase_de(Y),propiedades(P)),propiedadesc(Y,P1), append(P,P1,Z).

%Para consultar todas las clases representadas en los frames
clases(L):-findall(X,frame(X,subclase_de(_),propiedades(_)),L).

%Para consultar todas las subclases de una una clase
subclases_de(X,L):-findall(C1,subc(C1,X),L).

%Para consultar todas las superclases de una clase
superclases_de(X,L):-findall(C1,subc(X,C1),L).

%Para consultar qué objetos tienen UNA propiedad determinada
tiene_propiedad(P,Objs):-frame(X,_,propiedades(L)),member(P,L),subclases_de(X,S),select(X,Objs,S),

%Obtiene todas las propiedades de todos los objetos
todas_propiedades(L):-findall(P,frame(_,_,propiedades(P)),NL), flatten(NL,L).

% Obtiene una lista de clases con los objetos que tienen la propiedades
% de la lista de entrada en P
consulta_por_propiedades(P,C):-consulta(P,C1),list_to_set(C1,C2),sort(C2,C).

consulta([],[]).
consulta([H|T],C):-consulta(T,C1), tiene_propiedad(H,C2),append(C1,C2,C).

%Es hoja (regresa verdadero si la clase no tiene subclases)
```

El código en Python utiliza la biblioteca Tkinter para crear una interfaz gráfica que permite al usuario realizar consultas sobre la taxonomía de animales definida en Prolog.

Importación de Bibliotecas:

Se importan las bibliotecas necesarias, como Tkinter para la interfaz gráfica, Combobox para la lista desplegable, PySWIP para la conexión con Prolog y PIL para manejar imágenes.

Python.

Funciones de Consulta:

Se define una función consult_specie() para realizar la consulta a Prolog y mostrar las propiedades de la especie seleccionada en la interfaz gráfica.

```python
def consult_specie():
    global seleccionCombo, textResult

    if seleccionCombo == '':
        return

    prolog = Prolog()
    prolog.consult("marcos-animalia.pl")

    textResult.delete("1.0","end")

    propiedades_query = list(prolog.query(f"propiedadesc({seleccionCombo}, P)"))

    if propiedades_query:
        propiedades = propiedades_query[0]['P']

        if propiedades:

            for propiedad in propiedades:

                if "imagen" in propiedad:
                    a = propiedad.split("(")[1]
                    src = a.split(")")[0]
                    image2=Image.open(f"Imagenes/{src}")
                    image2 = image2.resize((150, 150),Image.Resampling.LANCZOS)
                    img_tk = ImageTk.PhotoImage(image2)
                    labelImg2 = Label(miFrame,image=img_tk)
                    labelImg2.grid(row = 4, column = 3, columnspan = 2)
                    labelImg2.image = img_tk

                else:
                    textResult.insert(END, propiedad+'\n')
        else:
            print("Advertencia", f"No se encontró información para {seleccionCombo}")
    else:
        print("Advertencia", f"No se encontró información para {seleccionCombo}")
```

Interfaz Gráfica:

Se crea la interfaz gráfica utilizando Tkinter, con etiquetas, una lista desplegable, un botón de consulta y un área de texto para mostrar los resultados.

```
root = Tk()
root.title('Taxonomia')

miFrame = Frame(root, width = 500, height = 500)
miFrame.pack(fill="both",expand=True)

bg=PhotoImage(file="Imagenes/salvaje.png")
my_label=Label(miFrame,image=bg)
my_label.place(x=0,y=0,relwidth=1,relheight=1)

tituloLabel = Label(miFrame, text = 'Taxonomia de animales', fg = "red", font = ("Arial", 29))

specieLabel = Label(miFrame, text="Introduzca la especie:",font=("Trebuchet MS",10))

especieCombo = Combobox(miFrame, values=animals)
especieCombo.bind("<<ComboboxSelected>>", mostrar_seleccion)

consultar = Button(miFrame, text="Consultar",width=17,command=consult_specie, bg = 'lime')

textResult = Text(miFrame, width=45,height=10)

image=Image.open("Imagenes/question.png")
image=image.resize((150,150),Image.Resampling.LANCZOS)

img=ImageTk.PhotoImage(image)
lbl_img=Label(miFrame,image=img)
```

Lista Desplegable y Consulta:

Se utiliza una lista desplegable (Combobox) para permitir al usuario seleccionar una especie. Se asocia la función consult_specie() al evento de selección.

```
especieCombo = Combobox(miFrame, values=animals)
especieCombo.bind("<<ComboboxSelected>>", mostrar_seleccion)
```

Mostrar Imágenes:

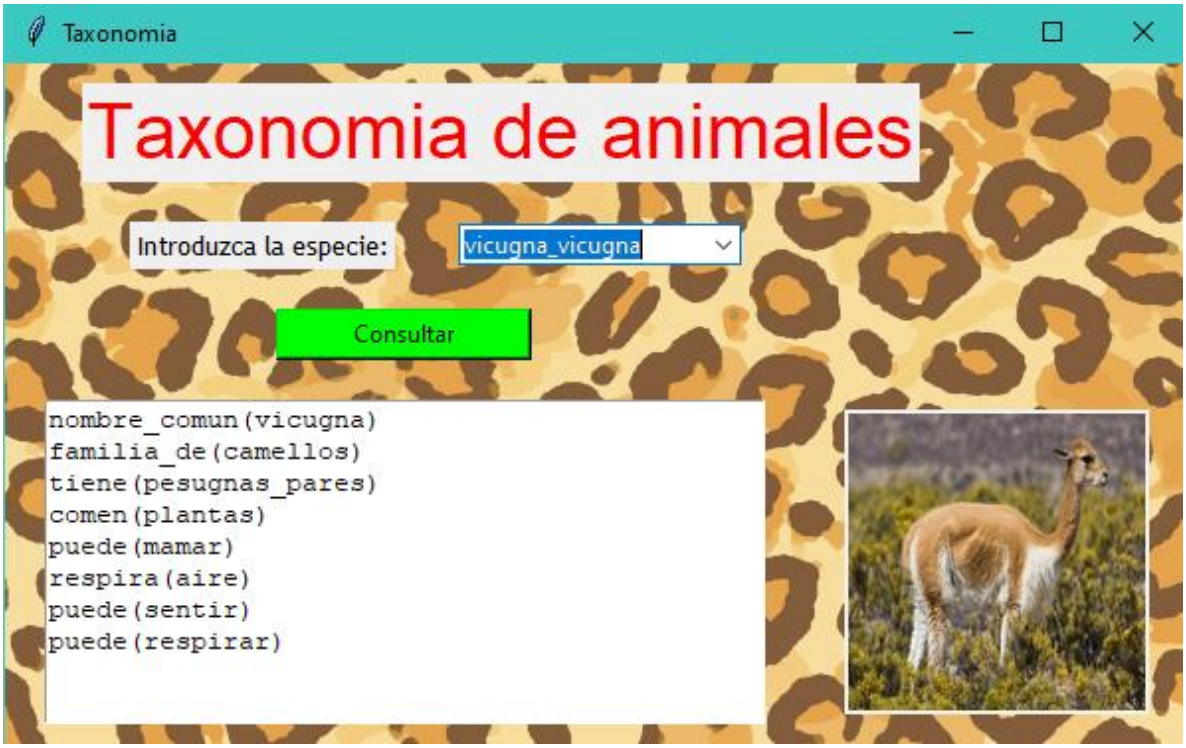
Se utiliza PIL para cargar y mostrar imágenes asociadas a ciertas propiedades en la interfaz gráfica.

```
def display_image(src):
    try:
        image2 = Image.open(f"Imagenes/{src}")
        image2 = image2.resize((100, 100), Image.Resampling.LANCZOS)
        img_tk = ImageTk.PhotoImage(image2)
        labelImg2 = Label(frame, image=img_tk)
        labelImg2.place(x=350, y=80)
        labelImg2.image = img_tk
    except FileNotFoundError:
        print(f"Imagen no encontrada: {src}")
```

DEMOSTRACIÓN DE FUNCIONAMIENTO

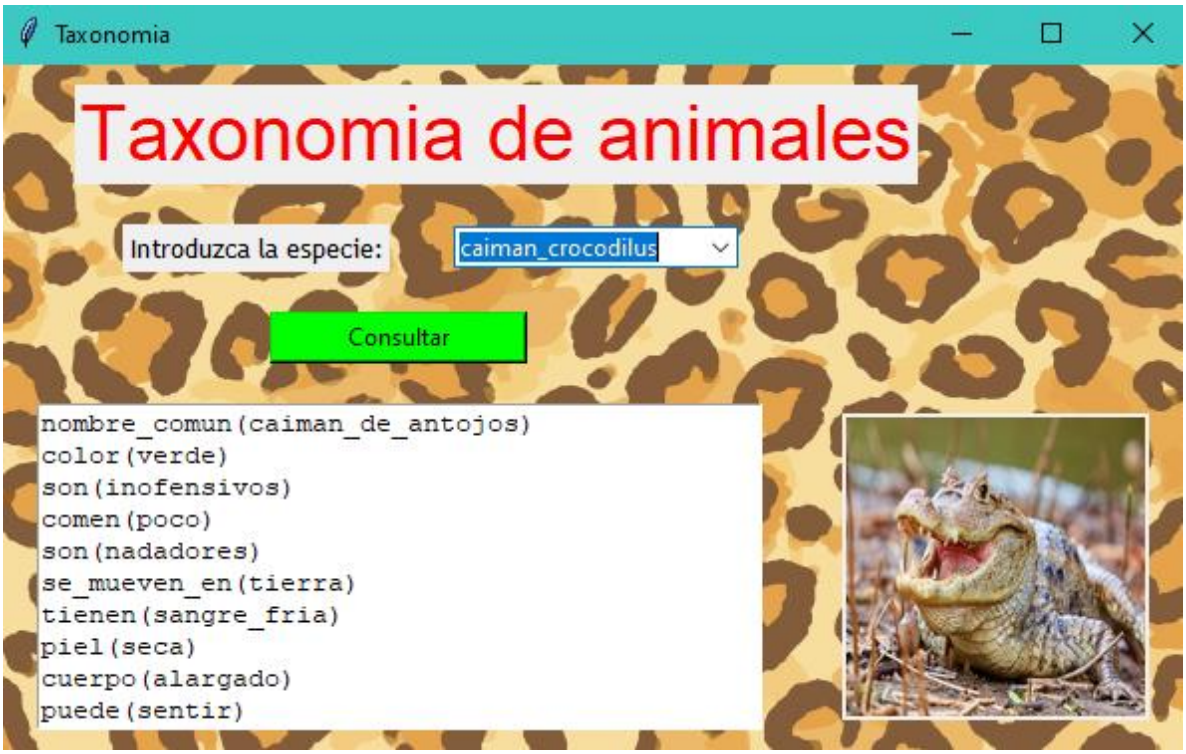
La pantalla principal del programa

Animal de la clase mamífero:

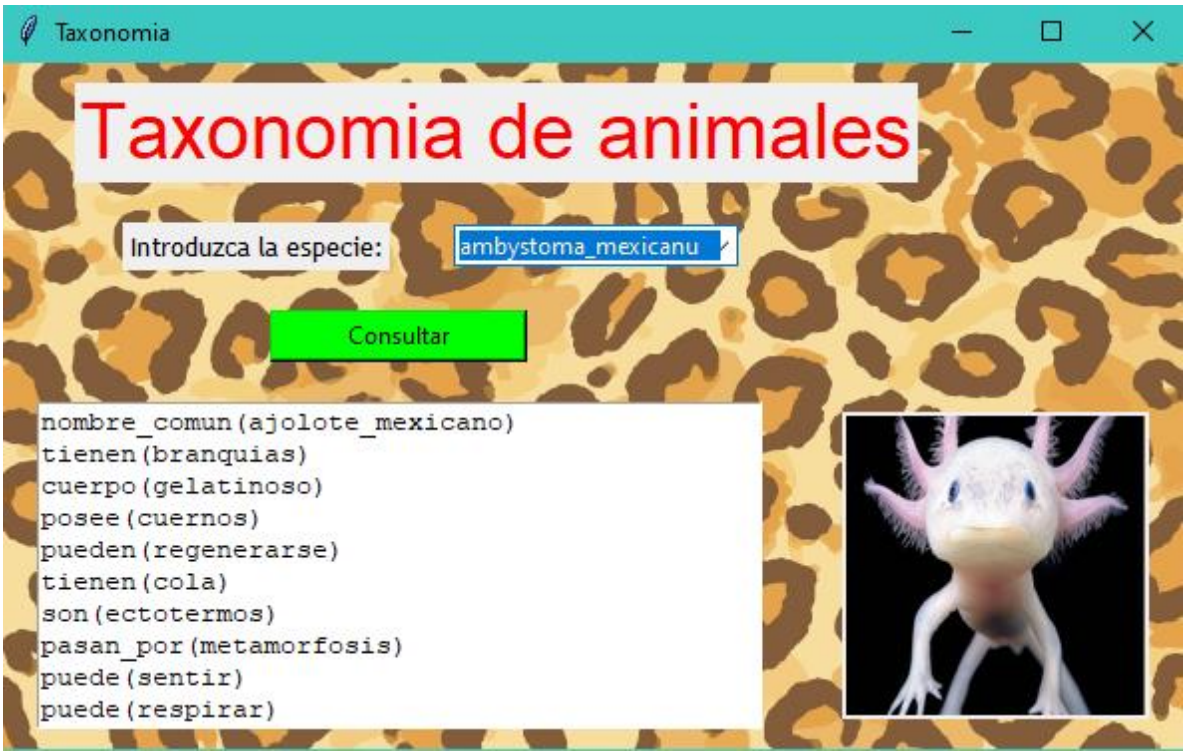


Animal de la clase ave:

Animal de la clase reptil:



Animal de la familia anfibio



Animal de la clase pez:

CONCLUSIONES

El trabajo realizado demuestra una implementación efectiva de la representación mediante marcos en Prolog para modelar la taxonomía de animales. La utilización de marcos y la jerarquía de clases permite organizar de manera estructurada y jerárquica la información sobre distintas especies animales. Además, la integración con Python mediante Tkinter y PySWIP proporciona una interfaz gráfica intuitiva que facilita la consulta de información específica sobre cada especie.

La implementación en Prolog refleja de manera clara y concisa la relación entre clases y subclases, así como la asignación de propiedades a cada categoría y especie. Las reglas de inferencia en Prolog son útiles para realizar consultas de manera eficiente, permitiendo explorar la taxonomía y obtener información detallada sobre las propiedades de las especies.

Etrabajo demuestra una implementación exitosa de la representación de conocimientos basada en marcos, combinando eficazmente Prolog y Python para crear una herramienta interactiva y educativa para explorar la taxonomía de animales. Este enfoque podría ser ampliado y aplicado en diversos contextos educativos y de investigación.

BIBLIOGRAFÍA

Tkinter Dialogs. (s. f.). Python documentation. https://docs.python.org/3/library/dialog.html

Lemos, G. (2020). AI with Python. *utn*.

https://www.academia.edu/42759181/AI_with_Python